Haust 2016

Þýðendur
T-603-THYD

Homework 2

Nemandi:
Steinn Elliði Pétursson

Kennitala:
250594-2759

3. október 2016

Kennari:
Friðjón Guðjohnsen

1

Consider the following grammar:

$S \to \mathbf{a}A|\mathbf{a}B|\mathbf{ac}A|\mathbf{ac}C$
$A \to A\mathbf{a}|BC$
$B \to B\mathbf{b}|\epsilon$
$C \to \mathbf{c}C|\mathbf{c}$

a

Construct a left-most derivation of the string **acbcca**. Show each step in the derivation.

Solution.

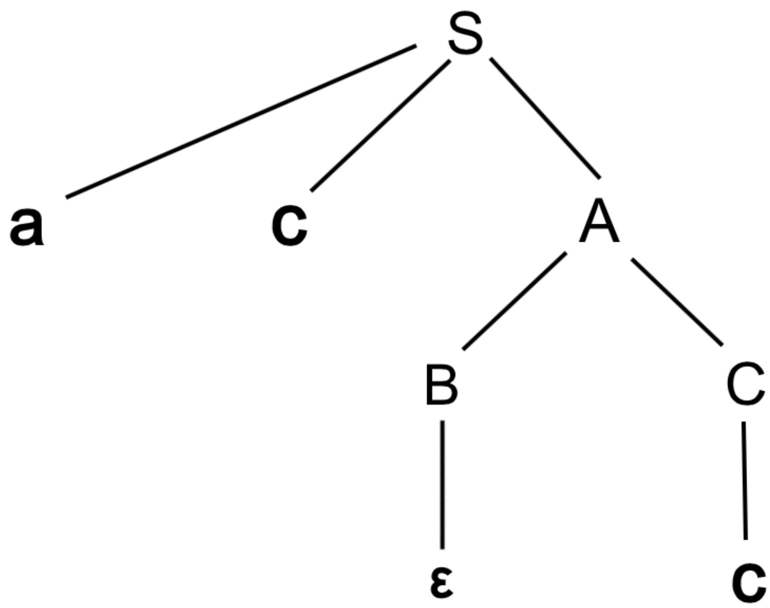
$S \to \mathbf{ac}A \to \mathbf{ac}A\mathbf{a} \to \mathbf{ac}BC\mathbf{a} \to \mathbf{ac}B\mathbf{b}C\mathbf{a} \to \mathbf{acb}C\mathbf{a} \to \mathbf{acbc}C\mathbf{a} \to \mathbf{acbcca}$

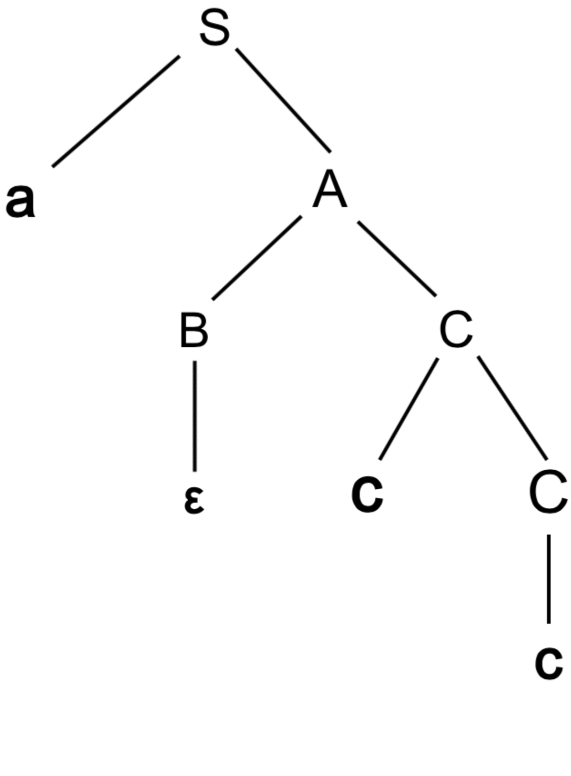
b

Is the grammar ambiguous? If you think that the grammar is ambiguous, then provide an example of a string for which two (or more) parse trees exist and show the parse trees. If you think that the grammar in unambiguous, then provide a convincing justification for your answer.

Solution.

This grammar is ambiguous. Following is an example of two parse trees for the string **acc**


S
a
c
A
B
C
ε
c


S
a
A
B
C
ε
c
C
c

c

Eliminate left-recursion from the grammar

Solution.

$S \to \mathbf{a}A|\mathbf{a}B|\mathbf{ac}A|\mathbf{ac}C$
$A \to BCA'$
$A' \to \mathbf{a}A'|\epsilon$
$B \to B'$
$B' \to \mathbf{b}B'|\epsilon$
$C \to \mathbf{c}C|\mathbf{c}$

d

Left factor the grammar you created in the previous step (c).

Solution.

$S \to \mathbf{a}|S'$
$S' \to A|B|\mathbf{c}S''$
$S'' \to A|C$
$A \to BCA'$
$A' \to \mathbf{a}A'|\epsilon$
$B \to B'$ $B' \to \mathbf{b}B'|\epsilon$
$C \to \mathbf{c}C'$
$C' \to C|\epsilon$

e

Construct a left-most derivation of the string acbcca, now using the grammar you created in the previous step (d).

Solution.

$S \to \mathbf{a}S' \to \mathbf{ac}S'' \to \mathbf{ac}A \to \mathbf{ac}BCA' \to \mathbf{acb}BCA' \to \mathbf{acb}CA' \to \mathbf{acbc}C'A' \to \mathbf{acbc}CA' \to \mathbf{acbcc}C'A' \to \mathbf{acbcc}A' \to \mathbf{acbcca}A' \to \mathbf{acbcca}$

2

Consider the following grammar:
$bexpr \to bexpr$ **or** $bterm|bterm$
$bterm \to bterm$ **and** $bfactor|bfactor$
$bfactor \to$ **not** $bfactor|($ $bexpr$ $)|$**true**$|$**false**
where the set of terminals is $\{\mathbf{not}, \mathbf{or}, \mathbf{and}, (,), \mathbf{true}, \mathbf{false}, \}$.

a

Eliminate left-recursion from this grammar

Solution.

bexpr $\rightarrow$ *bterm bexpr'*
bexpr' $\rightarrow$ **or** *bterm bexpr'*$|\epsilon$
bterm $\rightarrow$ *bfactor bterm'*
bterm' $\rightarrow$ **and** *bfactor bterm'*$|\epsilon$
bfactor $\rightarrow$ **not** *bfactor*$|$**(** *bexpr* **)**$|$**true**$|$**false**

b

Give the FIRST and FOLLOW sets for all non-terminal symbols in the grammar you created in the previous step (a).

Solution.

NON-TERMINAL	FIRST	FOLLOW
bexpr	{ not, (, true, false }	{), $ }
bexpr'	{ or, ϵ }	{), $ }
bterm	{ not, (, true, false }	{ or,), $ }
bterm'	{ and, ϵ }	{ or,), $ }
bfactor	{ not, (, true, false }	{ or,), $, and }

3

Consider the following LL(1) grammar:
$S \rightarrow \mathbf{a}AC$
$A \rightarrow BD$
$B \rightarrow \mathbf{bc}B|\epsilon$
$C \rightarrow \mathbf{c}|\epsilon$
$D \rightarrow \mathbf{d}$

a

Show the FIRST and FOLLOW sets for the non-terminals S, A, B, C and D.

Solution.

NON-TERMINAL	FIRST	FOLLOW
S	{ a }	{ $ }
A	{ b, d }	{ c, $ }
B	{ b, ϵ }	{ d }
C	{ c, ϵ }	{ $ }
D	{ d }	{ c, $ }

b

Construct a predictive parsing table for the given grammar.

Solution.

	a	b	c	d	$
S	$S \rightarrow \mathbf{a}AC$				
A		$A \rightarrow BD$		$A \rightarrow BD$	
B		$B \rightarrow \mathbf{bc}B$		$B \rightarrow \epsilon$	
C			$C \rightarrow \mathbf{c}$		$C \rightarrow \epsilon$
D				$D \rightarrow \mathbf{d}$	

c

Show the moves of a non-recursive predictive parser on the input **abcd**. Show each step, as done on page 228 in the textbook.

Solution.

Matched	Stack	Input	Action
	S$	abcd$	
	aAC$	abcd$	output $S \rightarrow aAC$
a	AC$	bcd$	match a
a	BDC$	bcd$	output $A \rightarrow BD$
a	bcBDC$	bcd$	output $B \rightarrow bcB$
ab	cBDC$	cd$	match b
abc	BDC$	d$	match c
abc	DC$	d$	output $B \rightarrow \epsilon$
abc	dC$	d$	output $D \rightarrow d$
abcd	C$	$	match d
abcd	$	$	output $C \rightarrow \epsilon$

4

Consider the grammar G given below:
$E \rightarrow E + E$
$E \rightarrow E * E$
$E \rightarrow (E)$
$E \rightarrow id$

a

Construct the collection of sets of LR(0) items for G, together with the DFA for viable prefixes that can appear on the stack of a shift/reduce parser.

Solution.

$E \to .E + E$
$E \to E. + E$
$E \to E + .E$
$E \to E + E.$
$E \to .E * E$
$E \to E. * E$
$E \to E * .E$
$E \to E * E.$
$E \to .(E)$
$E \to (.E)$
$E \to (E.)$
$E \to (E).$
$E \to .id$
$E \to id.$

b

Show the FIRST and FOLLOW sets for the nonterminal E.

Solution.

NON-TERMINAL	FIRST	FOLLOW
E	{ (, id }	{ \$, +, *,) }